Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (currently amended) A method of generating code to be deployed in an application server, comprising the steps of:

receiving an archive file to be deployed[[;]] , wherein the archive file is an Enterprise Java Bean archive file;

introspecting an input class included in the Enterprise Java Bean archive file to generate information relating to the input class;

generating a markup language description of the input class based on the generated information relating to the input class;

creating an event handler for method nodes found in the markup language description;

registering the event handler;

parsing the markup language description and invoking the registered event handler; and

generating output code using the invoked event handler.

2. (canceled)

3. (original) The method of claim 1, wherein the step of introspecting an input class included in the archive file comprises the steps of:

extracting information identifying methods included in the input class; and

for each method, extracting information relating to parameters of the method.

4. (original) The method of claim 3, wherein the step of generating a markup language description of the input class comprises the step of:

generating an Extensible Markup Language description of the input class based on the generated information relating to the input class.

5. (currently amended) The method of claim 4, wherein the step of creating an event handler comprises the step of:

creating a Simple Application Programming Interface for Extensible Markup Language event handler for a method node found in the markup language description.

6. (currently amended) The method of claim [[5]] 46, wherein the step of parsing the markup language description and invoking the registered event handler comprises the step of:

parsing the markup language description using a Simple Application Programming Interface for Extensible Markup Language parser and invoking the registered Simple Application Programming Interface for Extensible Markup Language event handler.

9

7. (original) The method of claim 1, wherein the step of creating an event handler

comprises the step of:

creating a plurality of event handlers for a method node found in the markup

language description.

8. (original) The method of claim 7, wherein the step of registering the event handler

comprises the step of:

registering each of the plurality of event handlers.

9. (original) The method of claim 8, wherein the step of parsing the markup language

description and invoking the registered event handler comprises the step of:

parsing the markup language description and invoking each of the plurality of

registered event handlers.

10. (original) The method of claim 9, wherein the step of generating output code

comprises the step of:

generating output code using each of the plurality of invoked event handler in

parallel.

11. (canceled)

12. (original) The method of claim 11, wherein the step of introspecting an input class

included in the archive file comprises the steps of:

extracting information identifying methods included in the input class; and

for each method, extracting information relating to parameters of the method.

13. (original) The method of claim 12, wherein the step of generating a markup language description of the input class comprises the step of:

generating an Extensible Markup Language description of the input class based on the generated information relating to the input class.

14. (currently amended) The method of claim 13, wherein the step of creating a plurality of event handlers comprises the step of:

creating a plurality of Simple Application Programming Interface for Extensible Markup Language event handlers for a method node found in the markup language description.

15. (currently amended) The method of claim [[14]] 50, wherein the step of parsing the markup language description and invoking the registered event handler comprises the step of:

parsing the markup language description using a Simple Application Programming Interface for Extensible Markup Language parser and invoking the plurality of registered Simple Application Programming Interface for Extensible Markup Language event handlers.

11

16. (currently amended) A system for generating code to be deployed in an application server comprising:

a processor operable to execute computer program instructions;

a memory operable to store computer program instructions executable by the processor; and

computer program instructions stored in the memory and executable to perform the steps of:

receiving an archive file to be deployed[[;]] , wherein the archive file is an Enterprise Java Bean archive file;

introspecting an input class included in the Enterprise Java Bean archive file to generate information relating to the input class;

generating a markup language description of the input class based on the generated information relating to the input class;

creating an event handler for a method node found in the markup language description;

registering the event handler;

parsing the markup language description and invoking the registered event handler; and

generating output code using the invoked event handler.

17. (canceled)

12

18. (currently amended) The system of claim 17, wherein the step of introspecting an

input class included in the archive file comprises the steps of:

extracting information identifying ~~systems~~ methods included in the input class;

and

for each ~~system~~ method, extracting information relating to parameters of the

~~system~~ method.

19. (original) The system of claim 18, wherein the step of generating a markup language

description of the input class comprises the step of:

generating an Extensible Markup Language description of the input class based

on the generated information relating to the input class.

20. (currently amended) The system of claim 19, wherein the step of creating an event

handler comprises the step of:

creating a Simple Application Programming Interface for Extensible Markup

Language event handler for a ~~system~~ method node found in the markup language

description.

21. (original) The system of claim [[20]] 47, wherein the step of parsing the markup

language description and invoking the registered event handler comprises the step of:

parsing the markup language description using a Simple Application

Programming Interface for Extensible Markup Language parser and invoking the

13

registered Simple Application Programming Interface for Extensible Markup Language event handler.

22. (currently amended) The system of claim 21, wherein the step of creating an event handler comprises the step of:

creating a plurality of event handlers for a ~~system~~ method node found in the markup language description.

23. (original) The system of claim 22, wherein the step of registering the event handler comprises the step of:

registering each of the plurality of event handlers.

24. (original) The system of claim 23, wherein the step of parsing the markup language description and invoking the registered event handler comprises the step of:

parsing the markup language description and invoking each of the plurality of registered event handlers.

25. (original) The system of claim 24, wherein the step of generating output code comprises the step of:

generating output code using each of the plurality of invoked event handler in parallel.

26. (canceled)

14

27. (currently amended) The system of claim 26, wherein the step of introspecting an input class included in the archive file comprises the steps of:

extracting information identifying ~~systems~~ methods included in the input class; and

for each ~~system~~ method, extracting information relating to parameters of the ~~system~~ method.

28. (original) The system of claim 27, wherein the step of generating a markup language description of the input class comprises the step of:

generating an Extensible Markup Language description of the input class based on the generated information relating to the input class.

29. (currently amended) The system of claim 28, wherein the step of creating a plurality of event handlers comprises the step of:

creating a plurality of Simple Application Programming Interface for Extensible Markup Language event handlers for a ~~system~~ method node found in the markup language description.

30. (currently amended) The system of claim [[29]] 51, wherein the step of parsing the markup language description and invoking the registered event handler comprises the step of:

parsing the markup language description using a Simple Application

Programming Interface for Extensible Markup Language parser and invoking the

plurality of registered Simple Application Programming Interface for Extensible Markup

Language event handlers.

31. (currently amended) A computer program product for generating code to be deployed

in an application server comprising:

a computer readable storage medium; and

computer program instructions, recorded on the computer readable storage

medium, executable by a processor, for performing the steps of:

receiving an archive file to be deployed[[;]] , wherein the archive file is an

Enterprise Java Bean archive file;

introspecting an input class included in the Enterprise Java Bean archive file to

generate information relating to the input class;

generating a markup language description of the input class based on the

generated information relating to the input class;

creating an event handler for a method node found in the markup language

description;

registering the event handler;

parsing the markup language description and invoking the registered event

handler; and

generating output code using the invoked event handler.

16

32. (canceled)

33. (currently amended) The computer program product of claim 32, wherein the step of

introspecting an input class included in the archive file comprises the steps of:

extracting information identifying ~~computer program products~~ methods included

in the input class; and

for each ~~computer program product~~ method, extracting information relating to

parameters of the ~~computer program product~~ method.

34. (original) The computer program product of claim 33, wherein the step of generating

a markup language description of the input class comprises the step of:

generating an Extensible Markup Language description of the input class based

on the generated information relating to the input class.

35. (currently amended) The computer program product of claim 34, wherein the step of

creating an event handler comprises the step of:

creating a Simple Application Programming Interface for Extensible Markup

Language event handler for a ~~computer program product~~ method node found in the

markup language description.

36. (currently amended) The computer program product of claim [[35]] 48, wherein the

step of parsing the markup language description and invoking the registered event

handler comprises the step of:

17

parsing the markup language description using a Simple Application

Programming Interface for Extensible Markup Language parser and invoking the

registered Simple Application Programming Interface for Extensible Markup Language

event handler.

37. (currently amended) The computer program product of claim 31, wherein the step of

creating an event handler comprises the step of:

creating a plurality of event handlers for a ~~computer program product~~ method

node found in the markup language description.

38. (original) The computer program product of claim 37, wherein the step of registering

the event handler comprises the step of:

registering each of the plurality of event handlers.

39. (original) The computer program product of claim 38, wherein the step of parsing the

markup language description and invoking the registered event handler comprises the

step of:

parsing the markup language description and invoking each of the plurality of

registered event handlers.

40. (original) The computer program product of claim 39, wherein the step of generating

output code comprises the step of:

generating output code using each of the plurality of invoked event handler in

parallel.

41. (canceled)

42. (currently amended) The computer program product of claim 41, wherein the step of

introspecting an input class included in the archive file comprises the steps of:

extracting information identifying ~~computer program products~~ methods included

in the input class; and

for each ~~computer program product~~ method, extracting information relating to

parameters of the ~~computer program product~~ method.

43. (original) The computer program product of claim 42, wherein the step of generating

a markup language description of the input class comprises the step of:

generating an Extensible Markup Language description of the input class based

on the generated information relating to the input class.

44. (currently amended) The computer program product of claim 43, wherein the step of

creating a plurality of event handlers comprises the step of:

creating a plurality of Simple Application Programming Interface for Extensible

Markup Language event handlers for a ~~computer program product~~ method node found in

the markup language description.

19

45. (currently amended) The computer program product of claim [[44]] 49, wherein the

step of parsing the markup language description and invoking the registered event

handler comprises the step of:

parsing the markup language description using a Simple Application

Programming Interface for Extensible Markup Language parser and invoking the

plurality of registered Simple Application Programming Interface for Extensible Markup

Language event handlers.

46. (new) The method of claim 5, wherein the step of registering the event handler

comprises the step of:

registering the created Simple Application Programming Interface for Extensible

Markup Language event handler for a method node found in the markup language

description.

47. (new) The system of claim 20, wherein the step of registering the event handler

comprises the step of:

registering the created Simple Application Programming Interface for Extensible

Markup Language event handler for a method node found in the markup language

description.

48. (new) The computer program product of claim 35, wherein the step of registering the

event handler comprises the step of:

20

registering the created Simple Application Programming Interface for Extensible Markup Language event handler for a method node found in the markup language description.

49. (new) The computer program product of claim 44, wherein the step of registering each of the plurality of event handlers comprises the step of:

registering the plurality of created Simple Application Programming Interface for Extensible Markup Language event handlers for a method node found in the markup language description.

50. (new) The method of claim 14, wherein the step of registering each of the plurality of event handlers comprises the step of:

registering the plurality of created Simple Application Programming Interface for Extensible Markup Language event handlers for a method node found in the markup language description.

51. (new) The system of claim 29, wherein the step of registering each of the plurality of event handlers comprises the step of:

registering the plurality of created Simple Application Programming Interface for Extensible Markup Language event handlers for a method node found in the markup language description.